Australian Government
Department of Defence
Defence Science and
Technology Organisation

Situation Awareness Logistics Tool (SALT) Forecasting Tool

Rodney Barabasz

Command and Control Division
Information Sciences Laboratory

DSTO-TN-0620

ABSTRACT

The Situation Awareness Logistics Tool (SALT) is a concept demonstrator developed under the Theatre Situation Awareness (TSA) task (JTW 02/212) for the J4 staff at HQJOC. SALT is a web application that enables theatre logisticians to access and share data on critical supplies, personnel, terrain and infrastructure pertinent to each operation. This document provides a brief design and implementation overview of SALT before going into depth regarding the incorporated Forecasting tool. The Forecasting Tool enables theatre logisticians to forecast the status of critical supplies for the near future using different assumptions about the potential demand on those supplies.

Published by

DSTO Information Sciences Laboratory
PO Box 1500
Edinburgh South Australia 5111 Australia

Telephone: (08) 8259 5555
Fax: (08) 8259 6567

AR-013-361
March 2005

APPROVED FOR PUBLIC RELEASE

Situation Awareness Logistics Tool (SALT) Forecasting Tool

Executive Summary

Currently there are no formal methods of forecasting the stock holdings at the theatre level. It relies on the individual logistician and their prior experience and knowledge to foresee any issues of potential concern. The SALT Forecasting Tool was designed to assist the logistician on this matter.

The Forecasting Tool is a proof of concept tool that draws upon the structured data in the Situation Awareness Logistics Tool (SALT) to offer an alternative visual representation, which adds temporal meaning to the data in order to provide greater Situation Awareness. The emphasis of this tool is to enable logisticians to view previous holding fluctuations in order to forecast supply holdings into the near future.

The tool was designed, developed and demonstrated to J4 Staff at Headquarters Joint Operations Command (HQJOC). It confirmed that such a tool was of value to logisticians. However the current reporting nature of stock holdings makes the problem extremely complex as the recording of levels can be inconsistent and irregular.

Contents

1. INTRODUCTION 1

2. SALT OVERVIEW 1
2.1 High Level Architecture 2
2.2 Data Flow 3
2.3 Data Model 4

3. SALT FORECASTING TOOL 6
3.1 Data Flow 7
3.2 Graphing Technologies 8
3.2.1 ColdFusion 8
3.2.2 Java Charting Packages 10
3.2.2.1 JClass ServerChart 10
3.2.2.2 JFreeChart 12
3.3 Graphical User Interface (GUI) 13

4. FORECASTING ALGORITHMS 15
4.1 Usage Rate 16
4.2 Simple Sawtooth Algorithm 17
4.3 Adaptive Smoothing Algorithm 18
4.4 Exponential Smoothing Algorithm 20
4.5 Double Exponential Smoothing Algorithm 21
4.6 Storage Capacity 22
4.7 Re-Order Point 22
4.8 Re-Order Quantity 22
4.9 Target Holding 23
4.10 Holding Limits 23
4.11 Critical Holding 23
4.12 Runout Time 24
4.13 Days Of Supply 24

5. POSSIBLE EXTENSIONS 25

6. DISCUSSION 26

7. CONCLUSION 26

8. REFERENCES 27

1. Introduction

On the Technological Aids sub task of the Theatre Situation Awareness (TSA) task (JTW 02/212) in the Command and Control Division (C2D), the main area of work has been to develop a concept demonstrator called the Situation Awareness Logistics Tool (SALT).

SALT was designed and developed with the J4 (Logistics) Branch at Headquarters Joint Operations Command (HQJOC), to enable them to:

- Share and view data on critical supplies, terrain, and military and civil infrastructure that is relevant to their needs.
- Fully understand what the data means with regard to sustaining operations. This could be done while planning (i.e. Course Of Action analysis) or monitoring operations.
- Predict what the status of critical supplies could be in the near future.
- Do all the above easily and rapidly on the current Joint Command Support System (JCSS) network.

The Forecasting Tool was added to the SALT concept demonstrator to predict the status of critical supplies. This report aims to give an insight into the design of the Forecasting Tool.

2. SALT Overview

The JCSS Standard Operating Environment (SOE) for Deployable Joint Force Headquarters (DJFHQ) and HQJOC includes Microsoft Internet Explorer 5.5 (IE 5.5). SALT was designed to use technologies that would give JCSS users access during demonstrations and trials of the SALT concept demonstrator through the use of only their web browser, without the need for deploying non-SOE compliant software on the JCSS network or individual desktop machines. This enables users of SALT to share and exploit information about geography, infrastructure, and stock holdings within an area of logistic interest across the JCSS network, which previously they could not easily do .

SALT contains tabular views for data entry and data analysis; a Map Tool identifying infrastructure, geographic features and the impact of time and space and modes of movement; and a Forecasting Tool for projecting the critical supplies based on expected usage rates.

2.1 High Level Architecture

Figure 1 shows the High Level SALT architecture.

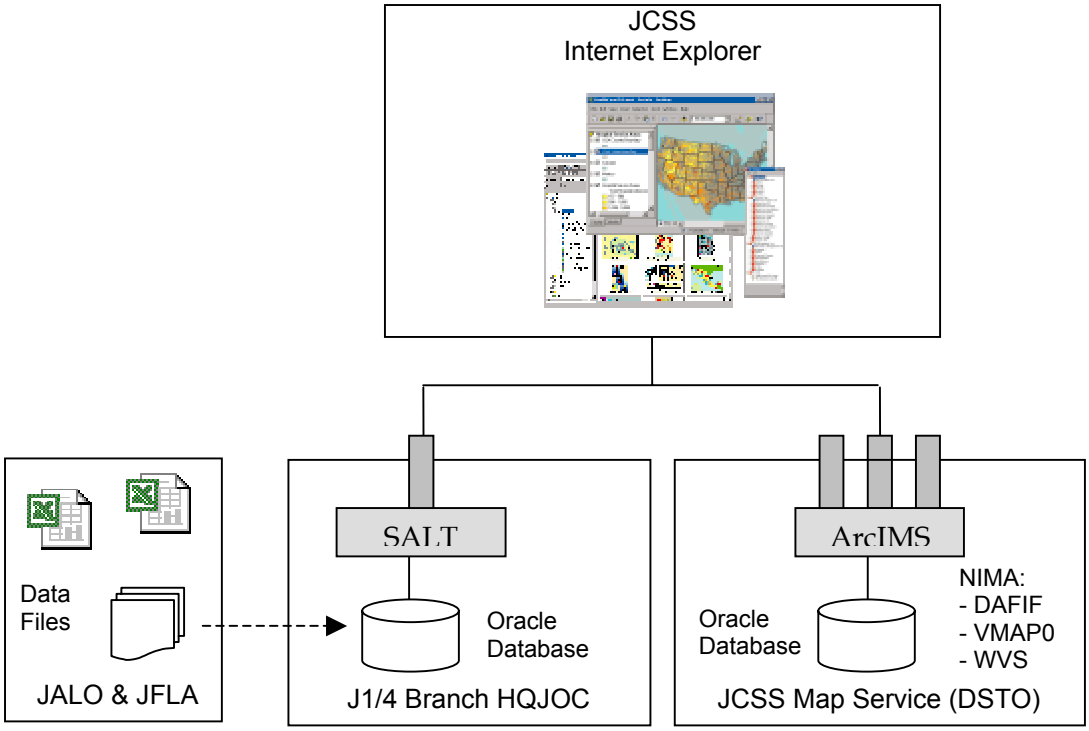

Figure 1 - SALT High Level Architecture

SALT obtains data on the types of munitions and their quantities and storage locations from the Joint Ammunition & Logistics Organisation (JALO) in the form of COMSARM (Computer System Armament) database dumps. SALT obtains fuel data from the Joint Fuels and Lubricants Agency (JFLA) in the form of Excel spreadsheets attached in emails on the Defence Secret Network (DSN). At present the COMARMS dumps and JFLA spreadsheets are entered manually to clean up the data prior to it being imported into the SALT database. It is envisaged that SALT would dynamically obtain data directly from those and other logistic information systems in the future.

The SALT infrastructure includes an Oracle Database, an Apache web server, and a Tomcat application server for executing Java Servlets and Java Server Pages (JSP). SALT's web interface allows users on JCSS to access the underlying SALT database to view and input data.

The SALT Map Tool requests map images from the ESRI ArcIMS package via a URL. The URL points to the ArcIMS Web Mapping Server (WMS) compliant map service and includes parameters for the part of map that is needed.

2.2 Data Flow

The data flow in SALT as shown in Figure 2 spans the client side and server side components. To provide a SALT user interface to the client, a complete cycle of the data flow needs to be traversed.

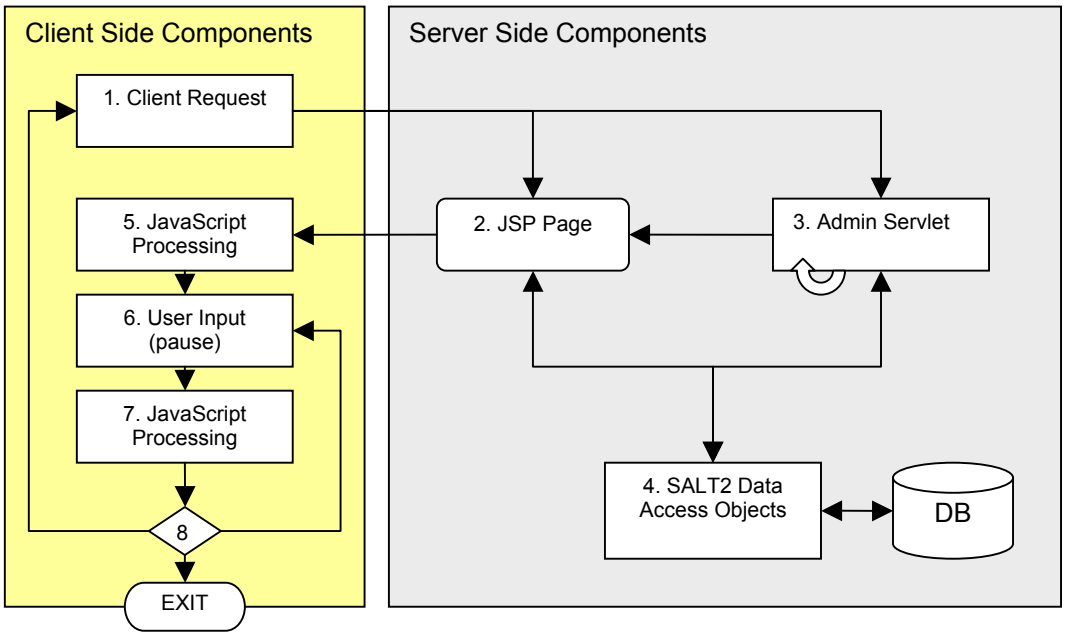

Figure 2 - SALT Data Flow Model

The data flow always begins with a client request (Fig. 2 – step 1). This is a HTTP 1.1 request to a SALT Java Server Page (JSP) or a Java Servlet. The request is made from the JCSS SOE default Web browser. Once a request has been received by the SALT server side components, further processing begins. JSP (Fig. 2 – step 2) and Servlet (Fig. 2 - step 3) resource requests are permissible. JSP requests usually result in the immediate display of output. Servlet requests are made to prepare data for the client and to bind to a persistent session to maintain client state. Once the requested Servlet completes its processing of the request, it is forwarded to the appropriate JSP.

A JSP resource renders the HTML output for a SALT web page. If no new data is required or the data is already present in the client's state, the JSP completes its processing and forwards the generated output back to the client. JSPs may use Data Access Objects (Fig. 2 – step 4), which are Java objects that perform further data processing; however this should not be necessary in most cases.

A Servlet resource usually performs data gathering and data processing. The primary use of the Servlets is to prepare the client state that is subsequently used by JSP components to render the output.

SALT JSP and Servlet resources are not limited to the functionality provided within SALT. They could make use of external resources through direct communication e.g. Java RMI (Remote Method Invocation). Alternatively they can be extended to access data from external web resources such as various web services.

Once the server side components complete their processing the result is returned to the caller, the web browser (Fig. 2 – step 5). Upon full receipt of the output, the client performs any JavaScript processing that is required. The JavaScript is part of the server side output and is generated by the JSPs. This step of processing is used to display error messages, and to populate dynamic interface components.

Once the JavaScript processing step completes, the SALT system pauses and waits for user input (Fig. 2 – step 6). The maximum wait time is defined in the SALT configuration and if it expires before the user makes any input, the user is timed out and an automated call to SALT is made to logout the client and terminate the session.

If user input is provided it is checked with JavaScript for validity. This processing (Fig. 2 – step 7) results in a prompt to the user to correct the input or the generation of another SALT request that begins a new SALT data cycle or an exit from the system.

2.3 Data Model

Figure 3 shows an Entity Relationship (ER) representation of the data model used within the SALT concept demonstrator.

SALT uses "Data Sheets" to identify sets of Holdings that are deemed by logisticians to be essential or critical to an operation. Data Sheets are similar to the Logistic Status reports (LOGSTATS) currently in use.

A Holding is the amount of an Item of Supply stored at a Facility. An Item of Supply can be anything from the following 10 Classes of Supply:

- *Class 1* – Rations and water
- *Class 2* – General stores including clothing, tentage, individual equipment, tool sets, and stationary.
- *Class 3* – Petroleum, oils and lubricants (*POL*).
- *Class 4* – Construction items and materials.
- *Class 5* – Ammunition.
- *Class 6* – Canteen supplies and non-scaled military items.
- *Class 7* - Principal items including vehicles, weapons and major technical equipments.
- *Class 8* – Medical and dental stores.

- *Class 9* – Repair parts.
- *Class 10* – Materials to support non-military programs.

Combat Supplies include Classes 1, 3, and 5. SALT pays special attention to the Combat Supplies as they are the most critical in any operation.

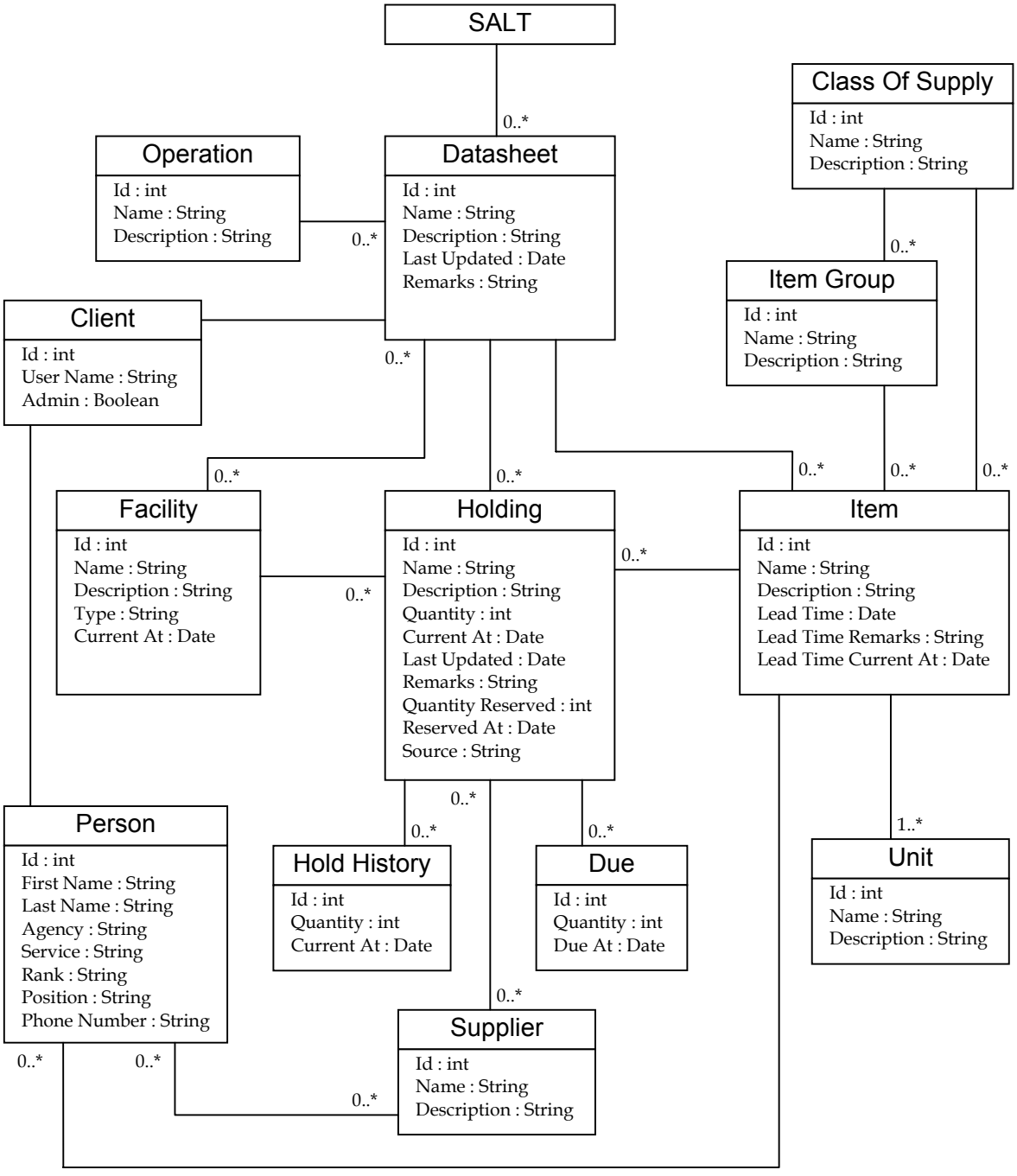

Figure 3 – SALT Data Model (simplified)

Classes of Supply are broken down into Item Groups for visualisation reasons, the main reason being that some Classes of Supply, such as Ammunition, have a large number of Items. Creating Item Groups such as "Missiles", "Bullets", "Grenades" etc for Class 3 and sorting by these Item Groups in a hierarchy, allows users to more easily find the Item of Supply they require. It also reduces the need for scrolling huge lists in the SALT User Interface.

A Facility is any entity where an Item of Supply can be stored or held, such as Ships, Airfields, Barracks and Bases. So an example of a Holding could be the amount of AVTUR (Aviation Turbine Fuel) held at RAAF Base Edinburgh.

A Holding has a Capacity associated with it (if known), a Current Holding, as well as a Holding History which is a list of previous current holding values. A Holding also has Dues associated with it. A Due can be a Due In, which is an expected arrival of more stock, or a Due Out which is an expected onward movement of stock to other facilities once a sufficient holding is available.

So a scenario might be "What quantities of AVTUR do we have in the AO (Area of Operations) for Operation *X*", where a corresponding Data Sheet represents the collection of facilities within that AO.

3. SALT Forecasting Tool

There are currently no formal methods for forecasting the stock holdings of supply items at the theatre level. It relies on the individual and their prior experience and knowledge to foresee any issues of potential concern. The SALT Forecasting Tool was designed to assist the individual on this matter.

The Forecasting Tool is a concept demonstrator that draws upon the structured logistics data in the SALT database to apply Forecasting Algorithms (described in Section 4) and represent the result in graphical form, thus providing greater Situation Awareness (SA) to the end users (the Logisticians at the theatre level). The nature of a concept demonstrator is to provide evidence that the approach is achievable and the concept is desirable; hence a variety of Forecasting Algorithms were implemented to demonstrate flexibility, whilst providing a comparison between the outcomes of each algorithm on the same set of data.

The following sections describe the investigation, design and implementation of the SALT Forecasting Tool.

3.1 Data Flow

The data flow in SALT for the Forecasting Tool is shown in Figure 4. To provide a Forecast Graph to the client, a complete cycle of the data flow needs to be traversed.

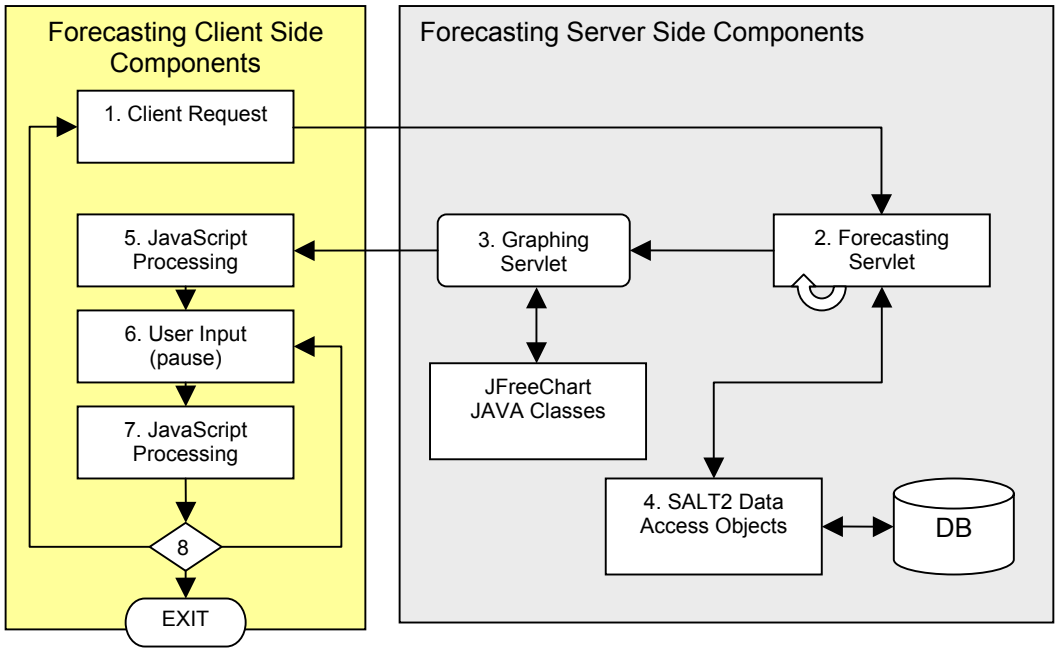

Figure 4 – Forecasting Data Flow Model

As described in Figure 2, the data flow begins with a client request (Fig. 4 – step 1). This is a HTTP request to the Forecasting Tool Java Servlet. The request is made from the JCSS SOE default Web browser. Once a request has been received by the tool server side components, further processing begins. Servlet (Fig. 4 – step 2) requests are made to prepare data for graphing by gathering the data and then traversing through the Forecasting algorithms to generate forecast data (See Section 4 for more details on the algorithms). Once the requested Servlet completes its processing the request is forwarded to the Graphing Servlet (Fig. 4 – 3).

The Graphing Servlet was specifically designed to be a generic graphing component which takes in the graphing dataset and uses a charting package to render an image output for a Forecast Tool web page. This implementation caters for possible re-use of the same graphing functionality in other areas of the SALT tool, not restricting its functionality to just the Forecasting Tool. It also allows for the implementation and comparison of various charting packages with minimal effects on the rest of the system.

The rest of the data flow cycle completes as described in Figure 2 (Section 2.2).

3.2 Graphing Technologies

Once the forecast algorithms were defined and the underlying architecture was in place, possible charting packages needed to be identified for use in the Forecasting Tool. When looking at charting packages there were specific goals in mind. The package had to be able to be instantiated in a web environment, as the SALT tool is a web application built around JavaScript, Dynamic HTML and Java Servlets. The charting package had to display multiple series of time-dependent data on one axis and allow for them to be represented as either a series of unrelated points or a line showing connectivity between points. This meant there was a need for the chart to have a time-dependent x-axis. The package also had to be able to handle irregular temporal data, as the SALT data may not contain regular values for each possible date within a graphing timeframe. Cost was also an issue.

Charting Packages take data input in a formatted form to produce a graph in an image form. Common methods of taking in data are by the use of the formatted eXtensible Mark-up Language (XML) according to the charting package-identified Document Type Definition (DTD) file, or by direct querying of a database, or by the use of programming language objects designed to store graphical data as defined by the package itself. XML-formatted data and package programming objects were acceptable for the Forecasting Tool. Direct querying of the database was not possible as code within the business rules layer of SALT was required when retrieving data from the SALT database to collect and validate the data.

The output of the charting package is a graph/chart in the form of an image that can usually be changed programmatically between JPEG/GIFF/PNG or even in some cases FLASH (an animated interactive image) and Scalable Vector Graphics (SVG). The emphasis was on keeping the image size down so the remote download times for the chart/graph were at a minimum.

The main charting alternatives considered were ColdFusion, and two Java Charting packages, JClass ServerChart and JFreeChart.

3.2.1 ColdFusion

(http://www.macromedia.com/software/coldfusion/)

ColdFusion allows creation of Web Applications using the ColdFusion Mark-up Language (CFML). CFML is a tag based language embedded into HTML pages that is used to execute server-side scripts. As shown in the code snippet in Figure 5, there is the tag named “cfchart” (the first word after the open bracket) and it has attributes “xAxisTitle” and “yAxisTitle”. There is also a “cfchartseries” tag which collects the data via a named query “DataTable” and displays it on the graph based on its defined attribute values.

Figure 5 – ColdFusion Example

ColdFusion charting features include direct querying of databases and the use of predefined XML structured graphing data.

ColdFusion met the requirements to a limited extent. Although only simple line graphs were required, those provided by ColdFusion were a little too primitive and restricted. The time-dependent graphs merely treated the date portion of the data as an x-axis label. This created problems as there was no grouping or sorting of date values, nor were there any differentials between time. There were workarounds like pre-sorting the data (by date and time), and ensuring that every possible date had a value so they were evenly spaced across the x-axis. This was a clumsy and inefficient solution.

ColdFusion had the advantage of being fully customisable. ColdFusion tags (more commonly known in other languages as components) could be customised, extended or created from scratch. The "cfchart" tag (as shown in Fig. 5) could have been extended to perform as the Forecasting Tool required, but a better alternative was to explore more appropriate packages. This avoided losing time exploring a new language, and developing a product that may have other unidentified restrictions.

ColdFusion requires the installation of the ColdFusion server on the Server side, which was an extra cost and burden on our infrastructure which was not desirable.

3.2.2 Java Charting Packages

Java charting packages can be bought (or even created from scratch) to extend a Java or non-Java application to enable it to display structured data in graph form. There are a number Java Chart packages available, most intended for web-based implementation. Some Internet resources list the numerous packages available and identify good and bad points about each. One resource is *http://www.programmersheaven.com/zone13/cat900/*

JClass Server Chart is a Commercial off-the-shelf (COTS) Java Charting package, and JFreeChart is an open source package. Both were implemented within the Forecasting Tool for assessment.

3.2.2.1 JClass ServerChart

(http://www.quest.com/jclass)

JClass allows a 30-day trial period of its ServerChart package. The trial can be downloaded from their website, and gives full functionality as well as all required developer documentation and Java Class libraries. However, it produces charts with a substantial watermark of "Evaluation" in various random colours.

JClass ServerChart is a large complex package. It caters for a wide range of implementation options and requires in depth knowledge of the package to obtain more than average results.

JClass have two licensing options with their ServerChart software. The Enterprise option is aimed at high level distributed applications with 1000 end-users and priced accordingly. This was not appropriate for the DSTO concept demonstrator model. The Developer licensing option is a pc/server dependent license for use by developers and 20 end-users. It retains the substantial watermark as does the evaluation license, reading "Development". This watermark can be seen in the Figure 6. The only way to create graphs without the obtrusive watermark is to purchase the costly Enterprise license.

Figure 6 was taken from the Forecast tool after the implementation of the JClass software. The horizontal lines at 500 and 1900 represent the Lower and Upper Limits respectively, which are user defined. The dashed line signifies that not all facility holdings are known at that time, and the solid line shows that all facility holdings are known. The squares show individual holding updates. Where there are no squares indicating Individual Holdings but there are deviations in the Total Holdings (represented by the deviating line), the deviations are caused by Dues In and Dues Out, which increase and decrease the Total Holdings respectively.

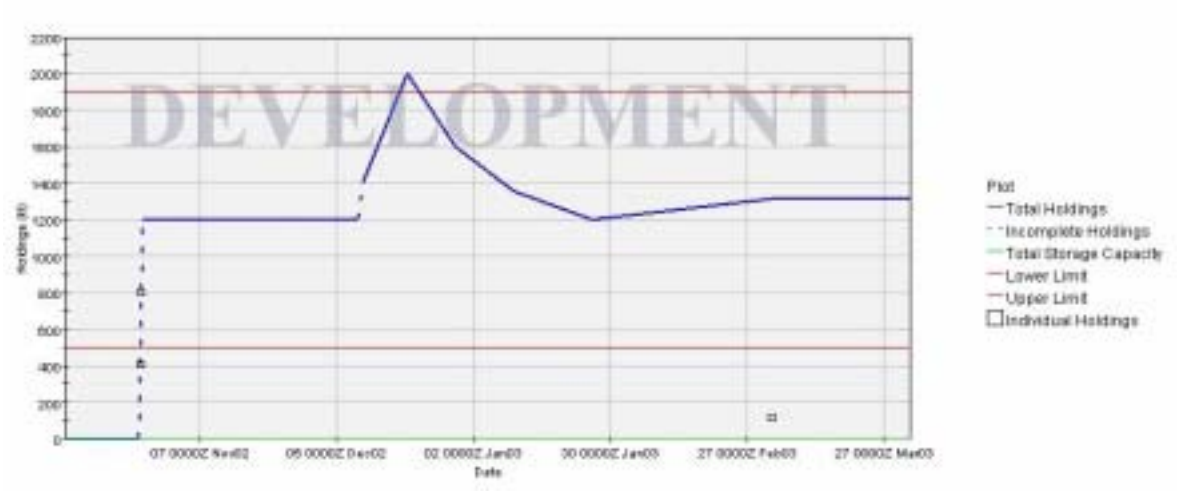

Figure 6 – JClass ServerChart Example.

JClass ServerChart components are Java Bean-compliant. This makes the creation of charts within JSP pages straightforward. All the usual chart layout and design parameters are available through the use of the Beans, however more complicated settings requires significant knowledge of the package and Java coding.

JClass ServerChart gives the developer the ability to supply chart data in XML format. It supplies its own Document Type Definition (DTD) file with the package, and allows the XML file to be passed into the charting package as a full dataset. There is also an additional XML option that allows the developer to supply chart layout and design parameters (through the use of another supplied DTD) in a different XML file. This allows the change of layout and design through the modification of this XML file without having to change and recompile code. This was somewhat cumbersome to create initially, but once created it was useful, especially if more than one type of graph was needed within the same application. This was achieved by setting up different layout and design XML files for each graph type to be displayed, and utilising the one that is required.

One of the requirements was to display the current time on the chart if it featured within the charting period. This charting package does not allow this. The closest alternative to meeting this requirement was to use the cross-hair symbol to indicate the current time on the x-axis. This was determined to be unsatisfactory the cross-hair was not very prominent.

ServerChart does allow different chart types on the same set of chart axis. This allows both points and series on the same graph as shown in Figure 10. It also caters for different predefined symbols for different chart point series, but it does not extend to

simple aspects like being able to fill shapes such as a solid square (only allows empty squares).

Scientific numbering on axes is also a problem with ServerChart. It lacked the ability to turn off scientific numbering or reformat it for greater readability.

Overall JClass ServerChart enabled the fast creation of simple charts; however it was significantly more cumbersome to create complex charts or to extend slightly on simple charts. JClass ServerChart is a very powerful package, but most options are not required for the Forecasting Tool.

3.2.2.2 JFreeChart

(http://www.jfree.org/jfreechart/index.html)

JFreeChart is a free open source Java class library for generating charts. The developer documentation for this package is not free, although it is inexpensive.

The source code for JFreeChart is also available under the GNU public License agreement. This allows alterations if the need arises. During the assessment of the JFreeChart package, modifications to the source code were only required on the one occasion, which was insignificant and scheduled to be included in later release of the package. There is also a forum for developers using JFreeChart on the main website. This is helpful in getting started, for development ideas, development problem solutions and feedback of potential problems with the package. The feedback from the forum is usually quick, accurate and most helpful.

JFreeChart is not easy to use. It requires considerable knowledge of the Java programming language as everything within a chart is customisable, even to creating shapes for points on the chart. It therefore takes longer (compared to the other charting packages) to set up a chart with the required look and feel. However, it does allow for more versatility. Figure 7 shows a snapshot of the Forecasting image after JFreeChart was implemented in the Forecasting Tool.

JFreeChart supports XML data sources. Although it has been flagged as a future enhancement, at the time of implementation the JFreechart XML Data Source did not cater for time-dependent data. This meant that JFreeChart Java data objects were used to store chart data and supply the data to chart.

At the time of implementation, JFreeChart did not directly cater for multiple chart types on the same axis; however because of its customisable nature the requirements were still achieved. For example, for the chart data where plot points were required, a display mechanism was created that would draw points and not draw lines, and vice-versa for the chart data required to be displayed as lines. Later releases of the package incorporated this as a standard facility.

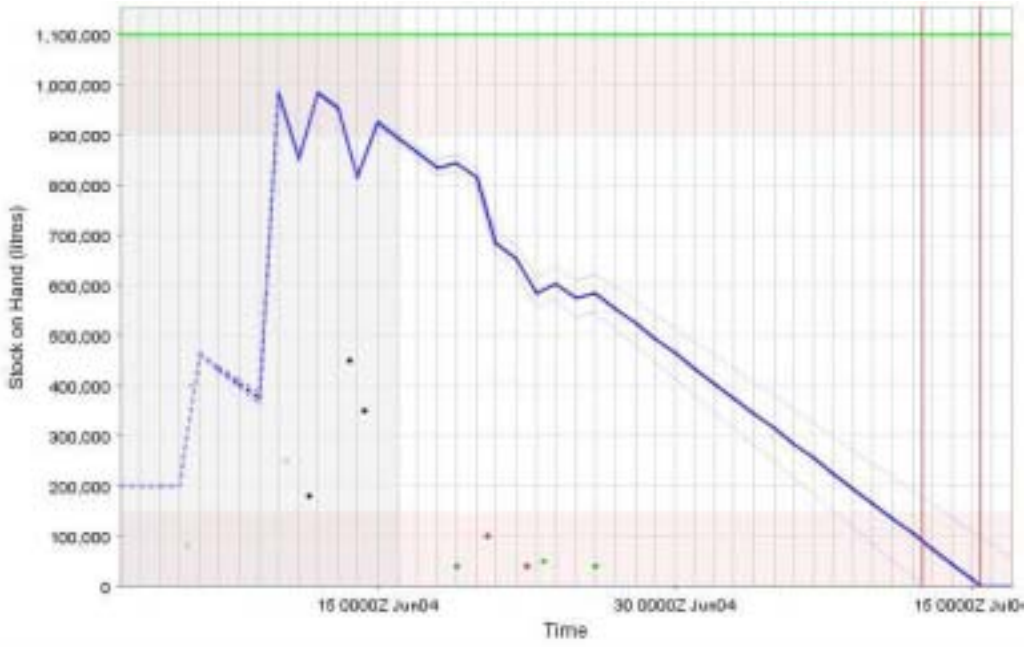

Figure 7 – JFreeChart Example.

Initially, there were complications with creating dashed line series in a chart. JFreeChart would start a new dash at each data point along that line, so if the data points were closer than the dash width, an undesirable solid line would be shown on the chart. A way around this could not be identified other than removing intermediate points that were collinear. A later release of the package rectified this problem (as shown in Figure 7).

Some minor problems were experienced with the labelling of the time axis. The axis labels and the data points are independent. This meant that if there is cyclic data every week, the weekly time labels would not conform to the cyclic data times. This is a known bug in JFreeChart and is scheduled to be fixed in future releases.

3.3 Graphical User Interface (GUI)

SALT spawns a new window for the Forecast Tool allowing the user to manipulate the Forecasting graph whilst still having a SALT screen available for monitoring and validating data. This also allows users to spawn multiple Forecast windows for various Items of Supply within the same Data Sheet or different Data Sheets.

Figure 8 shows a screen capture from the latest version of the SALT Forecasting Tool. The window opens to full screen size once the user has clicked on the Forecast button in the SALT user interface.

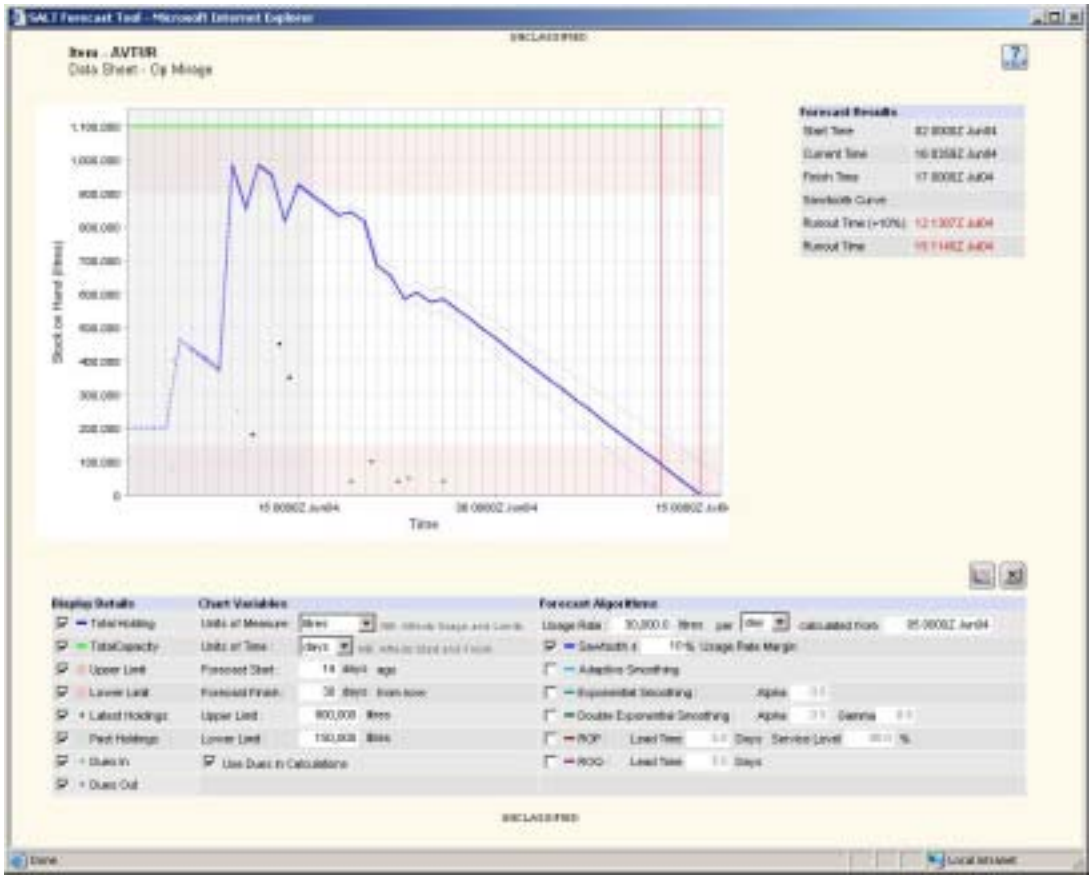

Figure 8 – Forecast Tool Screen Capture

The GUI of the Forecast Tool has a similar look and feel to the SALT interface even though it is a separate entity. This provides a familiar style of interface for the user, and thus is easier to learn and use.

As with SALT, the classification of the data is labelled at the top and bottom of the screen. The Item of Supply for Forecasting is labelled in the top left corner together with the name of the Data Sheet.

The middle section of the screen is taken up by the Forecast graph, which is a non-interactive image. The bottom of the Forecasting Tool contains a table of the controls for manipulating the graph.

The left column in the controls table, titled "Display Details", allows the user to toggle the display of various details in the graph by checking or un-checking the check boxes next to the name of the detail.

The middle column of the controls table, titled "Chart Variables", allows the user to manipulate the chart variables. The user can select the Units of Measure (y-axis) and the

Units of Time (x-axis) of the displayed chart. They can also select the forecasting graph timeframe. They simply enter in the number of Time Units they wish to display back in time, and the number of Time Units they wish to display forward in time from the current date. The Upper and Lower Limit values are user-entered, and are only allowed to be entered if the Upper and Lower Limit have been checked (in the left column of the controls table) to be displayed on the graph. It is possible to have Upper Limit displayed without a Lower Limit or vice-versa.

The right column of the controls table, titled "Forecast Algorithms", is for manipulation of the curves displayed on the graph. The user can enter a usage rate for the item, and then select which curve(s) they wish to display. The user enters additional parameters for each curve when the user selects that curve to be displayed, such as a percentage variation for the entered Usage Rate with the Sawtooth curve (as pictured). Hence the graph displays three curves as in Figure 8.

Displayed in the table to the right of the chart lists are the key results of the chart. The key results are sorted according to the data series which the results originated.

A default chart (using default chart variables) is derived once the forecast window is loaded. Once the user has made changes to the Chart Variables and visible Chart Features they wish to display, they can click on the chart button on the top right of the controls table to trigger a re-generation of the Forecast chart using the latest variables.

4. Forecasting Algorithms

Forecasting Problem statement:
Given a set of facilities and one Item of Supply, plot the Total Holding of that item for those facilities over a period of time into the near future.

While at first this may appear a simple requirement, a deeper analysis of the problem raises issues, such as:

- Holdings for some facilities may not be known.
- Holdings for different facilities can be reported at different times, and updates can be recorded at irregular intervals with days in between.
- An update for a holding may represent the usage at that individual facility; however the usage rate is not reported.
- An update for a holding may represent possible dues in and/or out; however it is not reported whether or what proportion of dues in or out have occurred.

Traditional forecast algorithms are based on a single time series i.e. measurements of a single quantity at regular time intervals. This does not match the problem as stated above. SALT data consists of multiple irregular time series (one from each facility) that are combined to be treated as a whole.

The following sections define and discuss the formulae used within the SALT Forecasting tool. They take into account (where possible) the previously mentioned issues relating to the forecasting problem.

4.1 Usage Rate

The Usage Rate is the amount of a particular Item of Supply that has been, or will be consumed during the conduct of an operation. The Usage Rate is calculated for a set of facilities rather than an individual facility, although a set can be a singleton.

Historical data can be used to determine Usage Rates. To do this, you could note decreases in the recorded holdings of the individual facilities. Consider the following example used by Brinschwitz and Surendonk [1], where the change over time is the average usage. If there is no known usage rate and no known delivery confirmations or amounts, the true day's usage is the drop in Holdings.

The following table shows an individual facility holding (or could be a combined facility holding) for POL (Petrol, Oil, and Lubricant) over a seven day period.

Day	1	2	3	4	5	6	7
POL Holding	1000	700	400	100	800	400	1200
Drop		300	300	300		400	

We can determine the usage over that seven day period by calculating the average drop in POL. The average drop is:

$$\frac{(300 + 300 + 300 + 400)}{7} = 229 \text{ litres per day for the seven days}$$

This average could have been reduced by small Dues In. A lift in tempo could have contributed to the consumption of an unusual amount of POL within the timeframe or the opposite where grounded aircraft could have contributed to a lower consumption.

With the SALT data, it is not always possible to distinguish between decreases solely caused by usage and decreases caused by a combination of usage and Dues In. It is also difficult to ascertain usage from certain facilities using the above method as the frequency in reported holdings may be too inconsistent and/or too infrequent.

When planning, logisticians may do what-if scenarios and so they need to manually enter a Usage Rate into the Forecasting Tool. Alternatively, other applications such as JOLTS (Joint Operation Logistic Tool Suite) could be used to estimate the usage rate.

4.2 Simple Sawtooth Algorithm

The Simple Sawtooth Algorithm, takes a constant Usage Rate and applies it consistently to the Total Holdings over time.

Individual holdings are often recorded at different times. This makes it difficult to accurately determine the total holding (the sum of known individual holdings) when taking into account the Usage Rate, Dues In and Dues Out. See Nelson [2] for the derivation of the equation.

Let

- t be the time for which we want to calculate the Total Holding. It can be any time in the past, present or future.
- $hold(f,i,t)$ be the amount of item i held at facility f as last known prior to and including time t.
- $hold(F,i,t)$ be the total holding of item i by the set of facilities F as estimated at time t.
- $holdTime(f,i,t)$ be the time when that holding was last updated prior to and including time t.
- $due(f,i,t)$ be an amount of item i that facility f expects to receive (if positive) or expects to supply (if negative) at some time between $holdTime(f,i,t)$ and time t.
- $usage(F,i,t)$ be the rate at which item i is being or expected to be consumed from the set of facilities F during the period leading up to the time t.
- $|F|$ be the number of facilities.
- Since $usage(f,i,t)$ is generally unknown, we approximate using $\frac{usage(F,i,t)}{|F|}$

$$hold(F,i,t) \approx \max\left(\begin{array}{l} 0, \\ \sum_{f \in F} hold(f,i,t) + \sum due(f,i,t) \\ -\frac{usage(F,i,t)}{|F|} \times \left(\sum_{f \in F} \left(t - holdTime(f,i,t)\right) \right) \end{array}\right)$$

Figure 9 shows the implementation of the Simple Sawtooth Algorithm. The dashed line represents the Total Holdings where not all holdings are known. The grey background area of the chart represents the past and the white background area of the chart represents the future. The projected Runout Time is the vertical line that appears when the Total Holdings reaches zero. Runout Time is described in Section 4.12. The dots in the future area of the chart represent expected Dues, whereas the dots in the history portion of the chart represent individual holding updates and holding history.

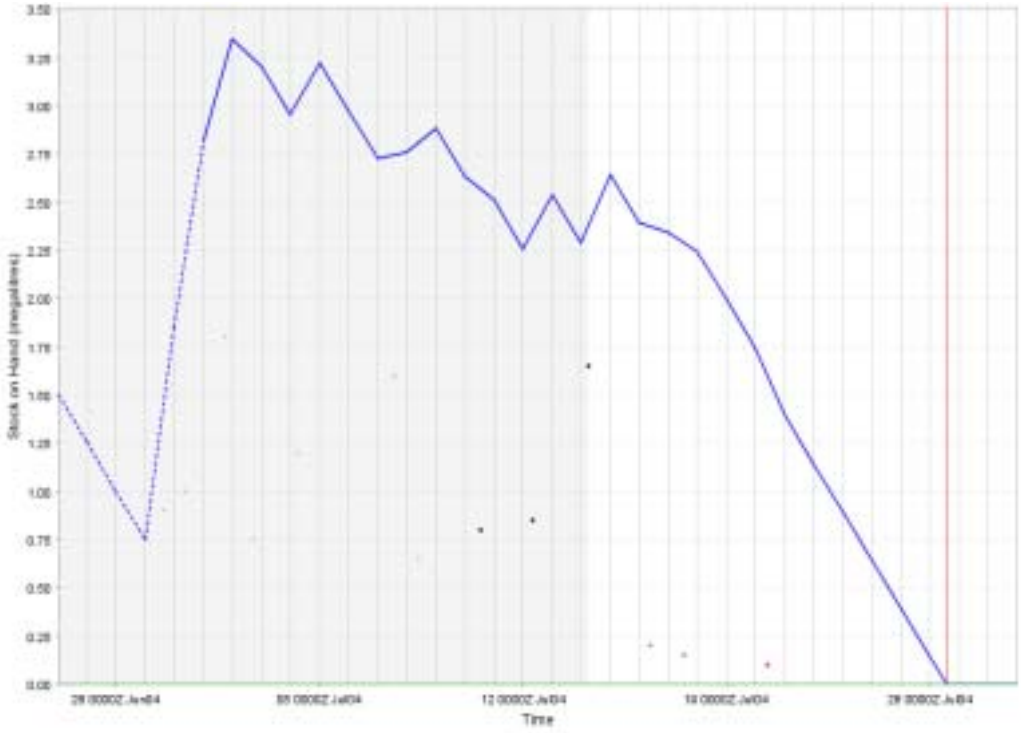

Figure 9 – Simple Sawtooth Algorithm

4.3 Adaptive Smoothing Algorithm

The Adaptive Smoothing Algorithm is a standard forecasting algorithm used in the Standard Defence Supply System (SDSS). This algorithm continually adjusts a variable called *alpha* on the basis of the current forecast error. If there is a large forecasting error, *alpha* will be large, until the forecast comes back on track. When the error is smaller, alpha will also be small and a stable forecast will result. See ASArmy [3] and ASArmy [4] for more information.

Let

- $alpha(n)$ be the alpha of the forecast period n.
- $fcsterr(n)$ be the Forecast Error of the forecast period n.
- $MAD(n)$ be the Mean Absolute Deviation, the approximate measurement of forecast accuracy for the forecast period n.
- $F(n)$ be the Adaptive Smoothing Forecast of the forecast period n.
- $Actual(n)$ be the Simple Sawtooth Algorithm (see section 3.1) as calculated for the forecast period n.

For the initial calculation, some initial conditions need to be set up as follows:

$$alpha(n-1) = 0.1$$
$$fcsterr(n-1) = 0$$
$$MAD(n-1) = \frac{abs[Actual(n) - Actual(n+1)]}{2}$$
$$F(n) = \frac{[Actual(n) + Actual(n+1))]}{2}$$

Then

$$fcsterr(n) = [0.1 \times (Actual(n) - F(n))] + [0.9 \times fcsterr(n-1)]$$
$$MAD(n) = [0.1 \times abs(Actual(n) - F(n))] + [0.9 \times MAD(n-1)]$$
$$alpha(n) = \frac{abs[fcsterr(n)]}{MAD(n)}$$
$$F(n+1) = [alpha(n-1) \times Actual(n)] + [1 - alpha(n-1)) \times F(n)]$$

Figure 10 compares the Simple Sawtooth Algorithm and the Adaptive Smoothing Algorithm.

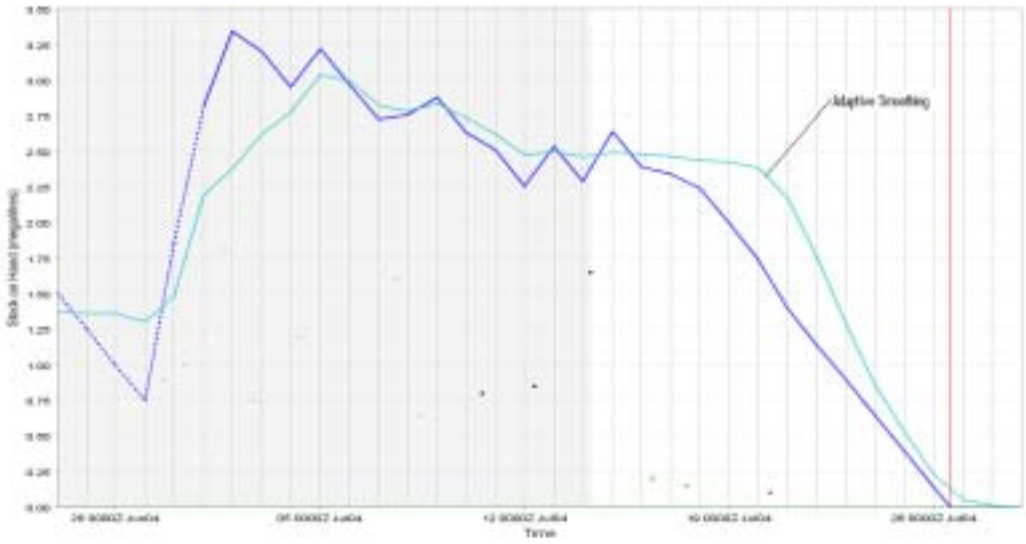

Figure 10 –Adaptive Smoothing Algorithm

Note the smoothing effects of the Adaptive Smoothing Algorithm to changes in Holdings, Dues In and Dues out.

4.4 Exponential Smoothing Algorithm

As with the Adaptive Smoothing Algorithm, the Exponential Smoothing Algorithm is also a standard forecasting algorithm used in the Standard Defence Supply System (SDSS). Exponential Smoothing is based on a simple idea that a new average can be computed from an old average and the most recent observed demand. Here, the variable *alpha* is a smoothing constant. See ASArmy [3] and ASArmy [4] for more information.

Let

- α be the alpha of the forecast period n. Where α is fixed and $0 \le \alpha \le 1$.
- x_t be the Exponential Smoothing value of the forecast period t.
- y_t be the Last Known Point of actual data.

$$x_{t+1} = (1-\alpha)x_t + \alpha y_t$$

Figure 11 compares the Simple Sawtooth Algorithm and the Exponential Smoothing Algorithm.

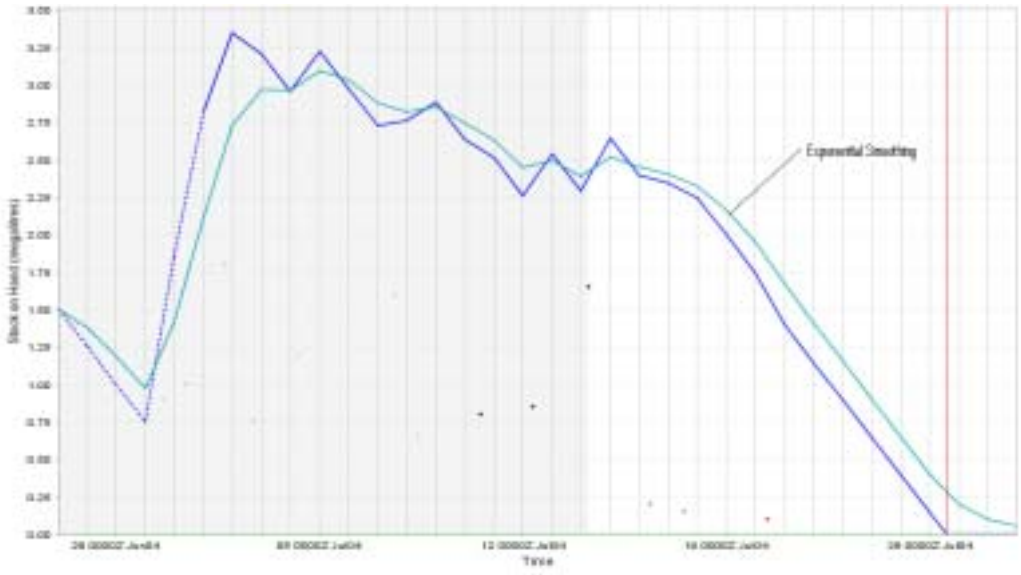

Figure 11 – Exponential Smoothing Algorithm

Note how the Exponential Smoothing Algorithm is more responsive to change in Holding, Dues In and Dues Out than the Adaptive Smoothing Algorithm.

4.5 Double Exponential Smoothing Algorithm

The Double Exponential Smoothing Algorithm is a standard forecasting algorithm used in the Standard Defence Supply System (SDSS). This algorithm eliminates the lag associated with the exponential smoothing algorithm by accounting for the trend. As with the Exponential Smoothing Algorithm, the *alpha* is a smoothing constant. This algorithm introduces a second constant (*gamma)* which is the slope-smoothing constant. See ASArmy [3] and ASArmy [4] for more information.

Let

- α be the alpha of the forecast period n, where α is fixed and $0 \le \alpha \le 1$.
- γ be the gamma of the forecast period n, where γ is fixed and $0 \le \gamma \le 1$.
- x_t be the Double Exponential Smoothing value of the forecast period t.

$$x_{t+1} = (1-\alpha)(x_t + b_{t-1}) + \alpha x_{t-1}$$
$$b_t = \gamma(x_t - x_{t-1}) + (1-\gamma)b_{t-1}$$

Figure 12 compares the Simple Sawtooth Algorithm and the Double Exponential Smoothing Algorithm.

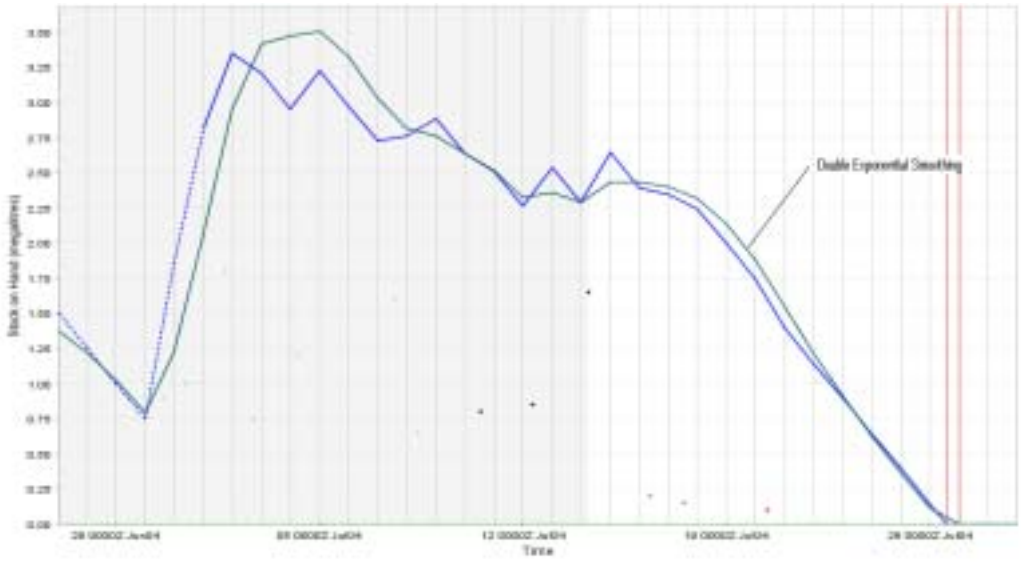

Figure 12 –Double Exponential Smoothing Algorithm

Note how the Double Exponential Smoothing Algorithm picks up trends in the data, causing it to overshoot, but then gradually re-adjust.

4.6 Storage Capacity

There are limits on how much of an Item of Supply can be held at an individual Facility. This limit is referred to generally as the Capacity for a Facility to hold an Item of Supply.

Let

- F be a set of Facilities that are of interest, say with regard to a given operation.
- i be a particular Item of Supply that is of interest.
- $cap(f,i)$ be the Capacity of an individual Facility f to hold item i.
- $cap(F,i)$ be the combined Storage Capacity for the set of Facilities F to hold item i.

$$cap(F,i) = \sum_{f \in F} cap(f,i)$$

4.7 Re-Order Point

Re-Order Point is the minimum stock value at which replenishing supply orders need to be placed. See ASArmy [3] for the detailed explanation and equation derivation.

Let

- ROP be the Re-Order Point for the forecast period.
- ANF be the Adjusted New Forecast for the forecast period.
- LT be the Lead Time of the Item of Supply.
- K be the K Factor in relation to the Service Level which is a constant. The Service Level is the percentage of demands which are to be satisfied on time and in full satisfaction.
- MAD be the Mean Absolute Deviation, the approximate measurement of forecast accuracy of the forecast period n. Calculation details for the MAD can be seen in section 3.4.
- 30.42 be the average number of days per calendar month.

$$ROP = \left[\frac{ANF}{30.42} \times LT \right] + (K \times MAD)$$

4.8 Re-Order Quantity

The Re-Order Quantity determines the level of stock ordered. See ASArmy [3] for the detailed explanation and equation derivation.

Let

- ROQ be the Re-Order Quantity for the forecast period.
- AMU be the Average Monthly Usage for the forecast period, where the AMU algorithm is that of the Adaptive Smoothing algorithm shown in section 3.4.
- LT be the Lead Time of the Item of Supply.

- 30.42 be the average number of days per calendar month.

$$ROQ = \left\lceil \frac{AMU}{30.42} \times LT \right\rceil$$

4.9 Target Holding

The Target Holding is the Total Holding determined over time that takes into account the requirements of the operation and provisions for contingencies while balanced against just-in-time supply that reduces the logistics footprint in the Area of Operations.

Logistics efforts therefore aim to achieve and maintain the Target Holding for each Item of Supply.

4.10 Holding Limits

The user may want to set limits on the Total Holding and be notified when those limits are transgressed. A Lower Limit is the minimum amount of an item that should be held by a set of stores at any one time. The maximum amount held by a set of stores at any one time is the Upper Limit.

4.11 Critical Holding

The Critical Holding is a Lower Limit of particular importance. If it is known how long it takes for supplies to arrive at a facility (or group of facilities) from resupply bases, then we can estimate the amount of holding necessary to sustain operations at the expected usage rate during that period of time. Thus if the Total Holding reaches the Critical Holding level and there are no supplies already on the way, then the set of stores may run out of the item in question, depending on the Usage Rate and any Dues In or Dues Out that may occur while waiting to be resupplied. See Nelson [2] for the equation derivation.

Let

- $holdCritical(F,i,t)$ be the critical holding for item i by the set of stores F as calculated at the time t.
- $LT(F,i,t)$ be the time it takes to resupply the set of stores F with item i as calculated at time t. Distance will obviously be a major factor in computing this time.
- $usage(F,i,t)$ be the rate at which item i is being or expected to be consumed from the set of facilities F during the period leading up to the time t.

$$holdCritical(F,i,t) = LT(F,i,t) \times usage(F,i,t)$$

4.12 Runout Time

The Runout Time is the time when a set of stores contains no more of the supply item in question. See Nelson [2] for the equation derivation.

Let

- $runoutTime(F,i,t)$ be the runout time for item i in the set of facilities F as calculated at the time t.
- $hold(f,i,t)$ be the amount of item i held at facility f as last updated before the time t.
- $usage(F,i,t)$ be the rate at which item i is being or expected to be consumed from the set of facilities F during the period leading up to the time t.
- $holdTime(f,i,t)$ be the time when that holding was last updated before the time t.
- $due(f,i,t)$ be an amount of item i that facility f expects to receive (if positive) or expects to transfer (if negative) at some time after $holdTime(f,i,t)$ and up to and including the time t.
- $|F|$ be the number of facilities.

The calculation time t will normally be the present time, but it could also be any time in the past or future. Choosing the calculation time effectively selects the Holding data, Dues In, Dues Out and Usage Rate to use in estimating the Runout Time.

$$runoutTime(F,i,t) = \frac{\sum_{f \in F} \max\left(0,\ hold(f,i,t) + \sum due(f,i,t)\right)}{usage(F,i,t)} + \frac{\sum_{f \in F} holdTime(f,i,t)}{|F|}$$

Where $usage(F,i,t) > 0$ and $|F| > 0$.

4.13 Days Of Supply

The Days of Supply is the number of days that a set of stores can continue to provide an item.

Let

- t_{dos} be the days of supply.
- $runoutTime(F,i,t)$ be the runout time for item i in the set of facilities F as calculated at the time t.

$$t_{dos} = runoutTime(F, i, t) - t \,.$$

The units of time are in days and the calculation time would normally be from the current time.

5. Possible Extensions

SALT has been developed as a concept demonstrator and the current Forecasting Tool reflects development time constraints. Therefore there remain a number of possible extensions to the existing Forecasting Tool.

At present the Upper and Lower limits are user entered values, each of which corresponds to a simple horizontal line drawn on the graph. These values could instead be calculated using various formulae depending on the Item of Supply (e.g. POL is not less than 10% of the capacity).

Alert states could be raised when the Total Holdings violates either the Upper or Lower limit. The Forecast Results table could then include items like Upper and Lower limit violations and Capacity violations.

Critical Holding (Section 4.11) and the Days of Supply (Section 4.13) are yet to be implemented in the current Forecasting Tool concept demonstrator. They could be implemented as an extension and displayed on the chart as well as the alert states displayed in the Forecast Results table.

The Forecast Chart could incorporate a drill-down capability. This would allow the user to point and click on the graph and receive feedback, whether a link back to the relevant stock holding pages in SALT or additional information. This would give the user a greater understanding of what is being displayed on the forecast graph.

The ability to create "What-if" scenarios was an original intention with the Forecasting tool. This did not eventuate due to time constraints, but would be a useful extension. The user would be able to modify the graphing data on the fly whilst viewing the Forecasting Chart without changing the underlying SALT data. This would enable the user to see the effects of such things as an extra Due In or Out, the loss of a critical holding due to the enemy combat operations, or the addition of an extra Stock Holding, etc. This would add great value to the Forecasting Tool. Additional functionality to save these "What-if" scenarios for reference in briefings would also be beneficial.

As mentioned earlier, the addition of various types of forecasting algorithms for different Items of Supply would add value to the Forecasting Tool. The ability to have

various algorithms (for user selection) for a single Item of Supply like Fuel would also add great value to the tool and reduce the knowledge currently required when entering in a manual usage rate.

6. Discussion

The potential of tools such as the SALT Forecasting Tool relies upon having access to reliable structured data. However, access to reliable data sources is often an issue with defence systems as many as these were designed as stove-pipe systems that do not easily cater for data sharing.

The heuristics used to select which forecasting algorithm to apply for a given Item of Supply for a desired level of supply (i.e. availability) in order to avoid over-stocking, and ensure availability of sufficient supplies when and where they are needed, are quite complex and require much experience. Some formulae used are mathematically inconsistent (i.e. time is added to a physical quantity in the ROP formula in section 4.1.7); however they produce reasonable forecasts when applied under specific conditions. This is a potential area for further research.

To support that research, statistical packages such as SPSS (Statistical Package for the Social Sciences), Matlab etc. could be used to investigate better forecasting and data analysis techniques. SPSS is a data management and statistical analysis product. It can perform a variety of data analysis and presentation functions, including statistical analyses and graphical presentation of data.

Various charting packages were explored in relation to their suitability for the SALT Forecasting Tool. The latest version of the Forecasting Tool utilises the open source JFreeChart package. The Commercial off-the-shelf (COTS) products that were trialled failed to provide the flexibility required within the budget available for the small scaled SALT concept demonstrator. However, it is recognised that a COTS product would be preferable in an operational implementation of the Forecasting Tool if maintainability and support from vendors is required, and this is not available with an open source package. A number of COTS products would produce the same results as was achieved with the JFreeChart package given budget for purchase and development time for tailoring and customisation.

7. Conclusion

There are currently no formal methods of forecasting the stock holdings at the theatre level. It relies on the individual logistician and their prior experience and knowledge to

foresee any issues of potential concern. The SALT Forecasting Tool was designed to assist the individual on this matter.

The SALT Forecasting Tool concept demonstrator draws upon the structured logistics data in the SALT database to apply Forecasting Algorithms and represent the result in a graphical form, thus providing greater Situation Awareness (SA) to the end users. As a concept demonstrator, the Forecasting Tool was expected to provide evidence that the approach is achievable and the concept is desirable.

The Forecasting Tool has demonstrated the added value from displaying data in chart form with functional added value from various formulae. It was demonstrated as part of the SALT tool to the J4 Branch at HQJOC and other relevant defence organisations. Feedback from the demonstration was positive, with the SALT tool combined with its underlying tools providing improved logistical Situation Awareness.

8. References

1. Brinschwitz, K & Surendonk, T (2002), *Bulk Liquid Tracking Tool,* TOB SOP
2. Nelson, M (2003), *Notes On Logistics Calculations,* DSTO C2D Informal Report
3. ASArmy (2003), Land Warfare Procedures – Combat Service Support LWP-CSS 4-1-6 *Provisioning in support of Land Operations*
4. ASArmy (1995), Army Logistic Instruction - Technical 10-2, SDSS Version 3 Procedures - *SDSS Inventory Control and Warehousing*

DISTRIBUTION LIST

Situation Awareness Logistics Tool (SALT) Forecasting Tool

Rodney Barabasz

AUSTRALIA

DEFENCE ORGANISATION

	No. of copies
Task Sponsor	
DGICD	1
S&T Program	
Chief Defence Scientist	shared copy
FAS Science Policy	
AS Science Corporate Management	
Director General Science Policy Development	
Counsellor Defence Science, London	Doc Data Sheet
Counsellor Defence Science, Washington	Doc Data Sheet
Scientific Adviser to MRDC, Thailand	Doc Data Sheet
Scientific Adviser Joint	1
Navy Scientific Adviser	1
Scientific Adviser - Army	1
Air Force Scientific Adviser	1
Scientific Adviser to the DMO M&A	1
Information Sciences Laboratory	
Chief Command & Control Division	Doc Data Sheet
Research Leader Command Decision Environments Branch	1
Research Leader Information Enterprises Branch	1
Research Leader Joint Command Analysis Branch	Doc Data Sheet
Research Leader Intelligence Information Branch	Doc Data Sheet
Head Human Systems Integration	Doc Data Sheet
Head Information Exploitation	Doc Data Sheet
Head Effects-Based Modelling and Analysis	Doc Data Sheet
Head Information Systems	Doc Data Sheet
Head Distributed Enterprises	Doc Data Sheet
Head Joint Operations Analysis and Support	Doc Data Sheet
Head Command Concepts and Architectures	Doc Data Sheet
Head Command Process Integration and Analysis	Doc Data Sheet
Head Intelligence Analysis	Doc Data Sheet
Task Manager, Dale Lambert	1
Author, Rodney Barabasz	1
DSTO Library and Archives	
Library Edinburgh	1 copy & Doc Data Sheet
Australian Archives	1

Capability Development Division

Director General Maritime Development	Doc Data Sheet
Director General Land Development	1
Director General Capability and Plans	Doc Data Sheet
Assistant Secretary Investment Analysis	Doc Data Sheet
Director Capability Plans and Programming	Doc Data Sheet
Director Trials	Doc Data Sheet

Chief Information Officer Group

Deputy CIO	Doc Data Sheet
Director General Information Policy and Plans	Doc Data Sheet
AS Information Strategy and Futures	Doc Data Sheet
AS Information Architecture and Management	Doc Data Sheet
Director General Australian Defence Simulation Office	Doc Data Sheet
Director General Information Services	Doc Data Sheet

Strategy Group

Director General Military Strategy	Doc Data Sheet
Director General Preparedness	Doc Data Sheet
Assistant Secretary Governance and Counter Proliferation	Doc Data Sheet

Navy

Maritime Operational Analysis Centre, Building 89/90	Doc Data Sheet &
Garden Island Sydney NSW	Distribution List (shared)
Deputy Director (Operations)	
Deputy Director (Analysis)	
Director General Navy Capability, Performance and Plans, Navy Headquarters	Doc Data Sheet
Director General Navy Strategic Policy and Futures, Navy Headquarters	Doc Data Sheet

Air Force

SO (Science) – Headquarters Air Combat Group, RAAF Base, Williamtown NSW 2314	Doc Data Sht & Exec Sum

Army

ABCA National Standardisation Officer	
Land Warfare Development Sector, Puckapunyal	e-mailed Doc Data Sheet
SO (Science), Deployable Joint Force Headquarters (DJFHQ) (L), Enoggera QLD	Doc Data Sheet
SO (Science) - Land Headquarters (LHQ), Victoria Barracks NSW	Doc Data & Exec Summ

Joint Operations Command

Director General Joint Operations	Doc Data Sheet
Chief of Staff Headquarters Joint Operations Command	Doc Data Sheet
Commandant ADF Warfare Centre	Doc Data Sheet
Director General Strategic Logistics	Doc Data Sheet
COS Australian Defence College	Doc Data Sheet

Intelligence and Security Group

DGSTA Defence Intelligence Organisation	1
Manager, Information Centre, Defence Intelligence Organisation	1 (PDF)
Assistant Secretary Capability Provisioning	Doc Data Sheet
Assistant Secretary Capability and Systems	Doc Data Sheet
Assistant Secretary Corporate, Defence Imagery and Geospatial Organisation	Doc Data Sheet

Defence Materiel Organisation

Deputy CEO	Doc Data Sheet
Head Aerospace Systems Division	Doc Data Sheet
Head Maritime Systems Division	Doc Data Sheet
Chief Joint Logistics Command	Doc Data Sheet

Defence Libraries

Library Manager, DLS-Canberra	Doc Data Sheet

OTHER ORGANISATIONS

National Library of Australia	1
NASA (Canberra)	1
State Library of South Australia	1

UNIVERSITIES AND COLLEGES

Australian Defence Force Academy	
Library	1
Head of Aerospace and Mechanical Engineering	1
Hargrave Library, Monash University	Doc Data Sheet
Librarian, Flinders University	1

OUTSIDE AUSTRALIA

INTERNATIONAL DEFENCE INFORMATION CENTRES

US Defense Technical Information Center	1 PDF
UK Dstl Knowledge Services	1 PDF
Canada Defence Research Directorate R&D Knowledge & Information Management (DRDKIM)	1
NZ Defence Information Centre	1

ABSTRACTING AND INFORMATION ORGANISATIONS

Library, Chemical Abstracts Reference Service	1
Engineering Societies Library, US	1
Materials Information, Cambridge Scientific Abstracts, US	1
Documents Librarian, The Center for Research Libraries, US	1

SPARES	5
Total number of copies: Printed: 32 PDF: 3	**35**

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION
DOCUMENT CONTROL DATA

1. PRIVACY MARKING/CAVEAT (OF DOCUMENT)

2. TITLE

Situation Awareness Logistics Tool (SALT) Forecasting Tool

3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED REPORTS THAT LIMITED RELEASE USE (L) NEXT TO DOCUMENT CLASSIFICATION)

Document (U)
Title (U)
Abstract (U)

4. AUTHOR(S)

Rodney Barabasz

5. CORPORATE AUTHOR

Information Sciences Laboratory
PO Box 1500
Edinburgh South Australia 5111 Australia

6a. DSTO NUMBER	6b. AR NUMBER	6c. TYPE OF REPORT	7. DOCUMENT DATE
DSTO-TN-0620	AR-013-361	Technical Note	March 2005

8. FILE NUMBER	9. TASK NUMBER	10. TASK SPONSOR	11. NO. OF PAGES	12. NO. OF REFERENCES
2004/1053373/1	JTW 02/212	DGICD	26	4

13. DOWNGRADING/DELIMITING INSTRUCTIONS

http://www.dsto.defence.gov.au/corporate/reports/ DSTO-TN-0620.pdf

14. RELEASE AUTHORITY

Chief, Command and Control Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved for public release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SA 5111

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS Yes

18. DEFTEST DESCRIPTORS

Computer applications
Theater level operations
Logistics information systems
Logistics planning
Situation awareness

19. ABSTRACT

The Situation Awareness Logistics Tool (SALT) is a concept demonstrator developed under the Theatre Situation Awareness (TSA) task (JTW 02/212) for the J4 staff at HQJOC. SALT is a web application that enables theatre logisticians to access and share data on critical supplies, personnel, terrain and infrastructure pertinent to each operation. This document provides a brief design and implementation overview of SALT before going into depth regarding the incorporated Forecasting tool. The Forecasting Tool enables theatre logisticians to forecast the status of critical supplies for the near future using different assumptions about the potential demand on those supplies.